A DiffServ-MPLS solution offering real-time end-to-end guarantees

Steven Martin, Pascale Minet, Laurent George

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A DiffServ-MPLS solution offering real-time end-to-end guarantees

Steven MARTIN — Pascale MINET — Laurent GEORGE

N° 4831

Mai 2003

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--4831--FR+ENG

A DiffServ-MPLS solution offering real-time end-to-end guarantees

Steven MARTIN* , Pascale MINET† , Laurent GEORGE‡

Thème 1 — Réseaux et systèmes
Projet HIPERCOM

Rapport de recherche n° 4831 — Mai 2003 — 27 pages

Abstract: In this paper, we are interested in providing deterministic end-to-end guarantees to real-time applications in the Internet. We focus on two QoS *Quality of Service* parameters: the end-to-end response time and the end-to-end jitter, parameters of the utmost importance for such applications. We propose a solution, very simple to deploy, based on a combination of DiffServ and MPLS. The *Expedited Forwarding* (EF) class of the *Differentiated Services* (DiffServ) model is well adapted for real-time applications as it is designed for flows with end-to-end real-time constraints. Moreover *MultiProtocol Label Switching* (MPLS), when applied in a DiffServ architecture, is an efficient solution for providing QoS routing. The results of our worst case analysis enable to derive a simple admission control for the EF class. Resources provisioned for the EF class but not used by this class are available for the other classes.

Key-words: DiffServ, MPLS, EF class, QoS, real-time constraints, deterministic guarantee, admission control, worst case end-to-end response time.

* Ecole Centrale d'Electronique, 53 rue de Grenelle,75 007 Paris, *steven.martin@ece.fr*
† INRIA Rocquencourt, Domaine de Voluceau, 78 153 Le Chesnay, *pascale.minet@inria.fr*
‡ Université Paris 12, LIIA, 120 rue Paul Armangot, 94 400 Vitry, *george@univ-paris12.fr*

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
Téléphone : +33 1 39 63 55 11 —Télécopie : +33 1 39 63 53 30

Une solution basée sur DiffServ et MPLS offrant des garanties temps-réel de bout-en-bout

Résumé : Dans ce rapport, nous nous intéressons aux garanties déterministes de bout-en-bout offertes aux applications temps-réel dans l'Internet. Nous considérons plus particulièrement les deux paramètres de qualité de service (QoS) les plus importants pour de telles applications : le temps de réponse de bout-en-bout et la gigue de bout-en-bout. Nous proposons une solution très simple à mettre en œuvre, basée sur DiffServ et MPLS. La classe EF (*Expedited Forwarding*) du modèle DiffServ (*Differentiated Services*) est bien adaptée pour les applications temps-réel car elle a été conçue pour des flux ayant des contraintes temps-réel de bout-en-bout. Par ailleurs, MPLS (*MultiProtocol Label Switching*) est une solution efficace pour le routage avec qualité de service dans une architecture DiffServ. Les résultats de notre analyse pire cas permettent de déduire un contrôle d'admission simple pour la classe EF. Les ressources attribuées à la classe EF mais inutilisées par cette classe sont disponibles pour les autres classes.

Mots-clés : DiffServ, MPLS, classe EF, QoS, contraintes temps-réel, garanties déterministes, contrôle d'admission, temps de réponse de bout-en-bout pire cas.

Contents

1 Introduction

New applications, particularly real-time applications (including voice and/or video), make the best-effort service model inadequate. Indeed, this model cannot provide Quality of Service (QoS) guarantees, in terms of bandwidth, delay, jitter or packet loss. To answer the end-to-end QoS requirements in networks, the IETF has first defined the *Integrated Services* architecture or IntServ [7]. This service model involves per flow signalling. Each router has to process signalling messages and to maintain a soft state per flow. The low scalability of the IntServ architecture led the IETF to define the DiffServ architecture [6]. This architecture is based on traffic aggregation in a limited number of classes. In particular, the *Expedited Forwarding* (EF) class has been proposed for applications requiring low end-to-end packet delay, low delay jitter and low packet loss (e.g. voice and video applications that are delay and jitter sensitive), the EF class being the highest priority class.

In addition, Internet Service Providers need traffic engineering to optimize the utilization of network resources and to improve performance of highly demanding traffics. *MultiProtocol Label Switching* (MPLS) has been introduced to improve routing performance. This technology is strategically significant for traffic engineering [3], especially when it is used with constraint-based routing. It requires route optimization based on a set of constraints, and route assignment. The former can be done by extending current routing protocols, like OSPF [1]; the latter has to perform label distribution and support explicit routing, like RSVP-TE [2]. As MPLS allows to fix the path followed by all packets of a flow, it makes easier the QoS guarantee. Moreover, a MPLS label can easily represent a DiffServ class of service.

In this paper, we propose a solution to guarantee deterministic end-to-end response times and jitters to all the EF flows in a DiffServ domain. Indeed, beyond the qualitative definition of the offered QoS, no deterministic end-to-end guarantees have been proved for the EF class. On the other hand, we show how the label switching technology can be used to implement our solution.

The rest of the paper is organized as follows. In section 2, we define the problem and the different models. Section 3 briefly discusses related work. The solution, given in section 4, is based on a combination of DiffServ and MPLS. Then, we show in section 5 how to compute upper bounds on the end-to-end response time and the end-to-end delay jitter of any flow belonging to the EF class, based on a worst case analysis. We then derive from this analysis a simple admission control defined in section 6. In section 7, we show how close the bounds we have found are to the real worst case end-to-end response times and jitters. For that purpose, we have designed a simulation tool that does an exhaustive analysis. Comparison results are illustrated by an example. Finally, we conclude the paper in section 8.

2 The problem

We investigate the problem of providing deterministic QoS guarantees, in terms of end-to-end response time and delay jitter, to any flow belonging to the *Expedited Forwarding* class in a DiffServ domain. The end-to-end response time and delay jitter of an EF flow are defined between the ingress node and the egress node of the flow in the considered domain. In this paper, we want to provide the guarantee that the end-to-end response time (resp. the delay jitter) of any flow belonging to the EF class will not exceed D (resp. J), where D and J are two parameters of the considered DiffServ domain. As we make no particular assumption concerning the arrival times of packets in the domain, the feasibility of a set of flows belonging to the EF class is equivalent to meet both constraints, whatever the arrival times of the packets in the domain.

Moreover, we assume the models defined in the following subsections.

2.1 DiffServ Model

In the DiffServ architecture [6], traffic is distributed over a small number of classes. Packets carry the code of their class. This code is then used in each DiffServ-compliant router to select predefined packet handling functions (in terms of queuing, scheduling and buffer acceptance), called *Per-Hop Behavior* (PHB). Thus, the network is divided in two parts: the nodes at the boundary of the network (ingress and egress routers), that perform complex treatments (packet classification and traffic conditioning) and the nodes in the core network (core routers), that forward packets according to their class code.

DiffServ provides coarser levels of service than IntServ due to flow aggregation, but is implementable in large networks, simplifies the construction of multi-domain services, and is well adapted for tariffing by class of service. Several per-hop behaviors have been defined:

- the *Assured Forwarding* (AF) PHB group [14]. Four classes, providing more or less resources in terms of bandwidth and buffers, are defined in the AF service. Each class manages three different drop priorities representing the relative importance of a packet in the class.
- the *Expedited Forwarding* (EF) PHB [15]. Traffic belonging to the EF service is delivered with very low latency and drop probability, up to a negotiated rate. This service can be used for instance by IP telephony.
- the *Best-Effort Forwarding* PHB is the default one.

2.2 Scheduling model

We consider that a DiffServ-compliant router implements Best-Effort, AF and EF classes. When a packet enters the node scheduler, it is scheduled with the other packets of its class waiting for processing. We consider that the scheduling of the EF class is FIFO. Moreover, we assume that packet scheduling is non-preemptive. Therefore, the node scheduler waits for the completion of the current packet transmission (if any) before selecting the packet having the highest priority.

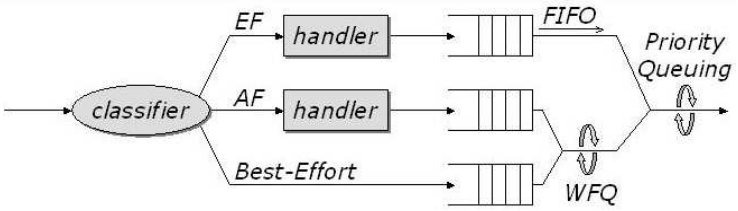

Figure 1: DiffServ-compliant router

As illustrated by figure 1, the EF class is scheduled with a *Fixed Priority Queuing* scheme with regard to the other classes. Thus, the EF class is served as long as it is not empty. Packets in the Best-Effort and AF classes are served according to *Weighted Fair Queuing* (WFQ) [18]. In this way, EF traffic will obtain low delay thanks to *Fixed Priority Queuing* scheduler and AF traffic will receive a higher bandwidth fraction than Best-Effort thanks to WFQ.

2.3 Network model

The considered network is a DiffServ domain (all routers are DiffServ-compliant). Moreover, all routers in the domain are Label Switching Routers (LSR). Links interconnecting routers are supposed to be FIFO and the transmission delay between two nodes has known lower and upper bounds: P_{min} and P_{max}.

Two DiffServ domain parameters are defined for the EF class:

- D, the worst case end-to-end response time guarantee,
- J, the worst case end-to-end delay jitter guarantee.

In this paper, we consider neither network failures nor packet losses.

2.4 Traffic model

We focus on the flows belonging to the EF class. We consider a set $\tau = \{\tau_1, ..., \tau_n\}$ of n sporadic flows belonging to the EF class. Each flow τ_i follows a sequence of nodes whose first node is the ingress node of the flow. In the following, we call line this sequence. Moreover, a sporadic flow τ_i is defined by:

- T_i, the minimum interarrival time (abusively called period) between two successive packets of τ_i;
- C_i^h, the maximum processing time on node h of a packet of τ_i;
- Jin_i^1, the maximum jitter of packets of τ_i arriving in the DiffServ domain.

This characterization is well adapted to real-time flows (e.g. process control, voice and video, sensor and actuator). Any flow $\tau_i \in \tau$ must meet the following constraints:

Constraint 1 *The end-to-end response time of any packet of τ_i from the ingress node to any egress node must not exceed D.*

Constraint 2 *When leaving a DiffServ domain, the difference between end-to-end response times experienced by any two packets of τ_i must not exceed J.*

For any packet g belonging to the EF class, we denote $\tau(g)$ the index number of the EF flow which g belongs to. Moreover, B^h denotes the maximum processing time at node h of any packet of flow not belonging to the EF class.

3 Related work

In this section, we first examine the existing approaches to obtain deterministic end-to-end response time guarantees in a multi-hop network. Then we see how these approaches can be applied to the Expedited Forwarding class of the DiffServ model. We then present works related to Diffserv and MPLS.

3.1 End-to-end response time

To determine the maximum end-to-end response time, several approaches can be used: a stochastic or a deterministic one. A stochastic approach consists in determining the mean behavior of the considered network, leading to mean, statistical or probabilistic end-to-end response times [21, 23]. A deterministic approach is based on a worst case analysis of the

network behavior, leading to worst case end-to-end response times [9, 12]. In this paper, we are interested in the deterministic approach as we want to provide a deterministic guarantee of worst case end-to-end response times for any EF flow in the network. In this context, two different approaches can be used to determine the worst case end-to-end delay: the holistic approach and the trajectory approach.

- The holistic approach [22] considers the worst case scenario on each node visited by a flow, accounting for the maximum possible jitter introduced by the previous visited nodes. If no jitter control is done, the jitter will increase throughout the visited nodes. In this case, the minimum and maximum response times on a node h induce a maximum jitter on the next visited node $h+1$ that leads to a worst case response time and then a maximum jitter on the following node and so on. Otherwise, the jitter can be either cancelled or constrained.

  - the Jitter Cancellation technique consists in cancelling, on each node, the jitter of a flow before it is considered by the node scheduler [11]: a flow packet is held until its latest possible reception time. Hence a flow packet arrives at node $h+1$ with a jitter depending only on the jitter introduced by the previous node h and the link between them. As soon as this jitter is cancelled, this packet is seen by the scheduler of node $h+1$. The worst case end-to-end response time is obtained by adding the worst case response time, without jitter (as cancelled) on every node;

  - the Constrained Jitter technique consists in checking that the jitter of a flow remains bounded by a maximum acceptable value before the flow is considered by the node scheduler. If not, the jitter is reduced to the maximum acceptable value by means of traffic shaping.

  In conclusion, the holistic approach can be pessimistic as it considers worst case scenarios on every node possibly leading to impossible scenarios.

- The trajectory approach [16] consists in examining the scheduling produced by all the visited nodes of a flow. In this approach, only possible scenarios are examined. For instance, the fluid model (see [18] for GPS) is relevant to the trajectory approach. This approach produces the best results as no impossible scenario is considered but is somewhat more complex to use. This approach can also be used in conjunction with a jitter control (see [10] for EDF, and [18] for GPS). In this paper, we adopt the trajectory approach without jitter control in a DiffServ domain to determine the maximum end-to-end response time of an EF flow.

We can also distinguish two main traffic models: the sporadic model and the token bucket model. The sporadic model has been used in the holistic approach and in the trajectory approach, while the token bucket model has been used in the trajectory approach.

- The sporadic model is classically defined by three parameters: the processing time, the minimum interarrival time and the maximum release jitter, (see section 2.4). This model is natural and well adapted for real-time applications.

- The token bucket [9, 10, 18] is defined by two parameters: σ, the bucket size, and ρ, the token throughput. The token bucket can model a flow or all flows of a DiffServ class. In the first case, it requires to maintain per flow information on every visited node. This solution is not scalable. In the second case, the choice of good values for the token bucket parameters is complex when flows have different characteristics. With this model, arises the problem of fixing the good values of these parameters for a given application. As shown in [10] and [20], the end-to-end response times strongly depend on the choice of the token bucket parameters. A bad choice can lead to bad response times. Furthermore, the token bucket parameters can be optimized for a given configuration, only valid at a given time. If the configuration evolves, the parameters of the token bucket should be recomputed on every node to remain optimal. This is not generally done.

3.2 Expedited Forwarding class

The definition of the EF PHB as given in [15] can be used to predict qualitative end-to-end delay guarantees. Beyond this definition, end-to-end guarantees are crucial for delay and jitter sensitive applications. Those QoS guarantees must be provided without requiring per flow processing in the core routers. Otherwise the solution would not be scalable. [5] shows that the worst case delay jitter for the EF traffic can be large in case of large networks.

The use of the FIFO scheduling algorithm for priority traffic in a network based on traffic aggregation (e.g. all flows in the EF class share a single FIFO queue) has been discussed in [8]. Nevertheless, the found delay bound is valid only for reasonably small EF traffic utilization.

In [13], a hybrid admission control scheme has been proposed, based on traffic shaping at border routers, to provide QoS guarantees in a DiffServ environment. The decision of the admission control is based on measurements realized to estimate resource allocation, leading to a higher utilization efficiency. As we are interested in deterministic guarantees, the admission control accounts for the worst case response times and jitters.

3.3 MPLS and DiffServ

MPLS forwards a packet based on its label, while IP routing forwards a packet based on its IP header, and notably the destination address. Label switching technology reduces forwarding delays due to simpler processing. Labels are carried in a *shim label header*, inserted either between the layer-2 and layer-3 headers, or directly in the layer-2 header. Moreover, while current IP routing does not guarantee that two packets of the same flow follow the same path between two endpoints, MPLS does. Label switching techniques with explicit routing capabilities allows traffic engineering by introducing the concept of a *traffic trunk* [3], which is a set of flows aggregated by their service class and then placed on a *Label Switched Path* (LSP).

At each LSR along an LSP, a packet is forwarded to the next hop based on its label. A solution for supporting the DiffServ Behavior Aggregates over a MPLS network is specified in [17]. Indeed, the 3-bit *Experimental* (Exp) field can encode, into the label header, the *Differentiated Services Code Point* (DSCP) which is used in each DiffServ-compliant router to select a predefined PHB. Since the DSCP requires six bits, two methods are proposed:

- if fewer than eight PHB are supported in the network, the Exp field is sufficient and its value can be mapped to the PHB. A LSP established in this way is called E-LSP;

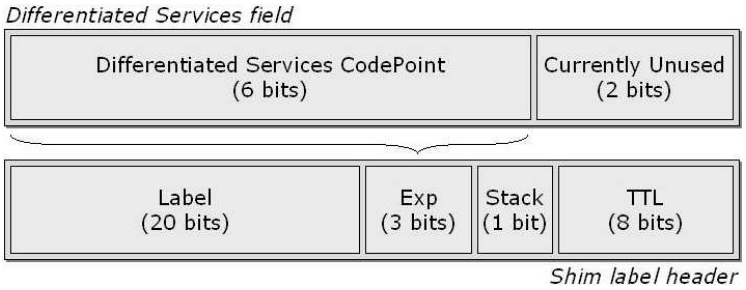

Figure 2: Encoding of the DSCP into the Exp field

- if more than eight PHB are supported in the network, the DSCP has to be encoded in the label. A LSP, called L-LSP when supporting such an approach, must be set up for each PHB class. In this case, the Exp field can provide the drop precedence of packets in the *Assured Forwarding* service class.

4 Proposed solution

For the problem defined in section 2, we propose a solution based on a combination of DiffServ and MPLS, where the EF class has the highest priority among the other classes. As at most eight classes are provided, the DSCP is encoded into the experimental field of the label. This solution is very simple for the following reasons:

- for a given destination, we need only one label. Hence the size of the label table is reduced;
- the QoS requirements are directly encoded into the label. Hence, the QoS routing is completely achieved at layer two, once the route has been established;
- each LSR uses a FIFO scheduling for the EF flows. FIFO scheduling is native in a DiffServ domain;
- the solution we propose requires neither to maintain information per EF flow nor to use traffic shaper in the core of the domain, leading to less complexity. The obtained solution can thus easily be implemented in a DiffServ domain.

The egress routers are in charge of ensuring that for any flow, the end-to-end jitter remains bounded by J. If it is not the case, the flow is reshaped to be compliant. This requires to synchronize egress router clocks to control the delay jitter at the end of the domain.

The upper bound on the end-to-end response time we give enables to derive a simple admission control. Indeed, the admission of a new flow τ_i in the EF class only requires a computation involving each node visited by τ_i. The conditions that have to be checked concern (i) the workload, (ii) the maximum sojourn time and (iii) the end-to-end response time of the EF flows visiting at least one node visited by τ_i. The solution with its associated admission control (see section 6) enables to ensure deterministic end-to-end response times for the EF flows in a DiffServ domain.

We now describe how a new EF flow is processed with our solution. When a new flow enters a DiffServ domain, the DSCP given in the DiffServ IP header determines the label appended to the packets of this flow. If this flow belongs to the EF class, the admission control is called. If accepted, the DiffServ domain guarantees the domain deadline D and jitter J, as defined in section 2.3. MPLS is then used for packet forwarding.

5 Worst case end-to-end real-time analysis

In this worst case analysis, we assume that time is discrete. [4] shows that results obtained with a discrete scheduling are as general as those obtained with a continuous scheduling when all flow parameters are multiples of the clock tick. In such conditions, any set of flows is feasible with a discrete scheduling if and only if it is feasible with a continuous scheduling.

In this section, we focus on the end-to-end response time of any packet m of any EF flow $\tau_i \in \{\tau_1, ..., \tau_n\}$ following a line $\mathcal{L}$, where $\mathcal{L}$ consists of q nodes numbered from 1 to q. We first introduce the notations and definitions used throughout the paper and define the constituent parts of the end-to-end response time of any EF flow. Then, we show how to compute upper bounds on the end-to-end response time of any flow belonging to the EF class, based on a worst case analysis of FIFO scheduling.

5.1 Notations and definitions

In the rest of this paper, we use the following notations and definitions:

e	element of the network such as a node, a line or the domain;
a_m^e	the arrival time of packet m in element e;
d_m^e	the departure time of packet m from element e;
R_m^e	the response time of packet m in element e;
S_m^h	the time taken by packet m to arrive on node h;
Jin_i^e	the worst case jitter of flow τ_i when entering element e;
$Jout_i^e$	the worst case jitter of flow τ_i when leaving element e;
$P_m^{h,h+1}$	the network delay experienced by packet m between nodes h and $h+1$.

Moreover, we denote $Rmin_i^e$ (resp. $Rmax_i^e$) the minimum (resp. the maximum) response time experienced by packets of flow τ_i in element e. Thus, we have: $\forall m$ of τ_i, $Rmin_i^e \leq R_m^e \leq Rmax_i^e$. In the same way, we denote $Smin_i^e$ (resp. $Smax_i^e$) the minimum (resp. the maximum) delay experienced by packets of flow τ_i from their arrival times in the domain to reach element e. Thus, we have: $\forall m$ of τ_i, $Smin_i^e \leq S_m^e \leq Smax_i^e$. Finally, as assumed in section 2.3, we have: $\forall m$ of τ_i, $Pmin \leq P_m^{h,h+1} \leq Pmax$.

Definition 1 *An idle time t is a time such that all packets arrived before t have been processed at time t.*

Definition 2 *A busy period is defined by an interval $[t, t')$ such that t and t' are both idle times and there is no idle time $\in (t, t')$.*

Definition 3 *For any node h, the processor utilization factor for the* EF *class is denoted U_{EF}^h. It is the fraction of processor time spent by node h in the processing of* EF *packets. It is equal to $\sum_{i=1}^{n}(C_i^h/T_i)$.*

5.2 Constituent parts of the end-to-end response time

Let us consider any EF flow τ_i, $i \in [1,n]$, following a line $\mathcal{L}$, where $\mathcal{L}$ consists of q nodes numbered from 1 to q. The end-to-end response time of any packet m of τ_i depends on its sojourn times on each visited node and network delays. Thus, we have:

$$R_m^{\mathcal{L}} = \sum_{h=1}^{q} R_m^h + \sum_{h=1}^{q-1} P_m^{h,h+1}.$$

As detailed in section 2.2, packet scheduling is non-preemptive. Hence, whatever the scheduling algorithm in the EF class and despite the highest priority of this class, a packet from another class (i.e. Best-Effort or AF class) can interfere with EF flows processing due to non-preemption. Indeed, if any EF packet m enters node $h \in [1,q]$ while a packet m' not belonging to the EF class is being processing on this node, m has to wait until m' completion.

Generally, the non-preemptive effect is not limited to this waiting time. For example, EF packets may arrive in the node while m is waiting for processing (due to m'). If they have a higher priority than packet m, then they will be processed before m. But these packets would not be necessarily processed before m if m' did not exist. The resulting delay incurred by m is indirectly due to the non-preemption. As we consider in this paper that the scheduling of the EF class is FIFO, no EF packet has priority over m if it enters the node after m did. Hence, we get the following property.

Property 1 *The maximum delay due to packets not belonging to the EF class incurred by any packet of the* EF *flow τ_i is bounded by: $\sum_{h=1}^{q}(B^h - 1)$, where B^h denotes the maximum processing time at node h of any packet of flow not belonging to the* EF *class.*

Proof: Let us show that if m is delayed on any node h of line $\mathcal{L}$ by a packet m' not belonging to the EF class, then in the worst case the packet m arrives on node h just after the execution of m' begins. Indeed, by contradiction, if m arrives on the node before or at the same time as m', then as m belongs to the EF class and this class has the highest priority, m will be processed before m'. Hence a contradiction. Thus, in the worst case, m arrives on node h one time unit after the execution of m' begins, because we assume a discrete time. The maximum processing time for m' on h is B^h. By convention, if all packets visiting node h belong to the EF class, we note $B^h - 1 = 0$. Hence, in any case, the maximum delay incurred by m on h directly due to a packet not belonging to the EF class is equal to $B^h - 1$. As we make no particular assumption concerning the classes other than the EF class, in the worst case packet m is delayed for $B^h - 1$ on any visited node h. ■

Obviously, the execution of packet m on any node $h \in [1,q]$ can also be delayed by packets belonging to the EF class and following a line $\mathcal{L}'$ with $\mathcal{L}' \bigcap \mathcal{L} \neq \emptyset$. If we denote X_{EF} this maximum resulting delay, then the worst case end-to-end response time of packet m of flow τ_i is bounded by:

$$R_{max}^{\mathcal{L}}{}_m \leq X_{EF} + \sum_{h=1}^{q} (B^h - 1) + (q-1) \cdot P_{max}.$$

In the following, we show how to compute the end-to-end response time of any packet m of flow τ_i, based on a worst case analysis. This analysis enables to evaluate X_{EF}, the delay due to other EF packets.

5.3 Determination of the end-to-end response time

In this section, we show how to compute the worst case end-to-end delay for an EF packet m as it traverses the DiffServ domain. The strategy is to move backwards through the sequence of nodes m traverses, each time identifying preceding packets and busy periods that ultimately affect the delay of m. We proceed in that way till finally at node 1, the very first packet $f(1)$ to affect m's delay is identified. Thereafter, the worst case delay is estimated from the time that $f(1)$ enters the domain to the time that m exits the domain. This, when subtracted by the difference between the arrival times of m and $f(1)$ to the domain, yields the worst case delay for m.

5.3.1 All the EF flows follow the same sequence of nodes

We assume in this subsection that all the EF flows follow the same line $\mathcal{L}$ consisting of q nodes numbered from 1 to q. This assumption allows to clearly illustrate the strategy adopted to determine the worst case end-to-end response time of any EF flow. In subsection 5.3.2, we show how to generalize this result.

Decomposition of the end-to-end response time of m

Let us consider any packet m belonging to any EF flow $\tau_i \in \tau$. To compute the end-to-end response time of packet m, we identify the busy periods that affect the delay of m. For this, we consider the busy period bp^q in which m is processed on node q and we define $f(q)$ as the first EF packet. If packets not belonging to the EF class are processed in bp^q before m, $f(q)$ is the first EF packet after the last of those packets. If no such packet exists, $f(q)$ is defined as the first packet of bp^q.

The packet $f(q)$ has been processed in the busy period bp^{q-1} on node $q-1$. We then define $f(q-1)$ as the first EF packet processed in bp^{q-1} and after the last packet not belonging to the EF class. If no such packet exists, $f(q-1)$ is the first packet of bp^{q-1}. And so on until the busy period of node 1 in which the packet $f(1)$ is processed (see figure 3).

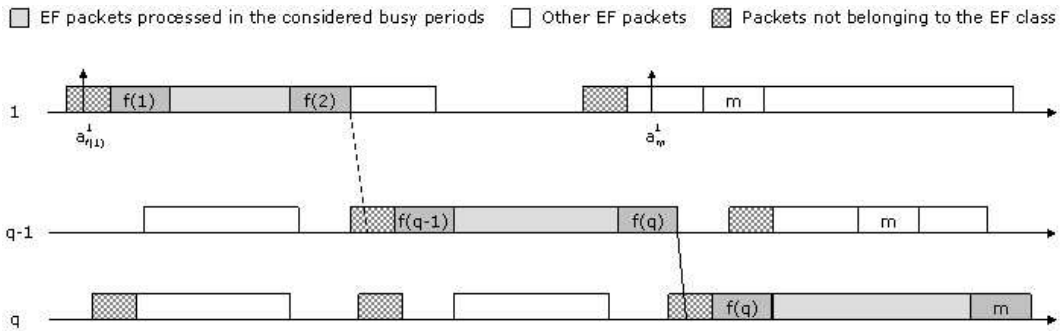

Figure 3: Response time of packet m

In the worst case, on any node $h \neq 1$, packet $f(h)$ arrives on node h one time unit after the beginning of bp^h. For the sake of simplicity, on a node h, we number consecutively the EF packets entering the domain between the arrival times $a^1_{f(1)}$ and a^1_m of $f(1)$ and m respectively (with $f(q+1) = m$). Hence, on node h, we denote $m'-1$ (resp. $m'+1$) the packet preceding (resp. succeeding) m'. By adding parts of the considered busy periods, we can now express the latest departure time of packet m from node q, that is:

$$\begin{aligned} a^1_{f(1)} &+ (B^1 - 1) + \text{the processing time on node 1 of packets } f(1) \text{ to } f(2) + P^{1,2}_{f(2)} \\ &+ (B^2 - 1) + \text{the processing time on node 2 of packets } f(2) \text{ to } f(3) + P^{2,3}_{f(3)} \\ &\ldots \\ &+ (B^q - 1) + \text{the processing time on node } q \text{ of packets } f(q) \text{ to } m. \end{aligned}$$

The end-to-end response time of packet m is then bounded by:

$$a^1_{f(1)} - a^1_m + \sum_{h=1}^{q} \left(\sum_{g=f(h)}^{f(h+1)} C^h_{\tau(g)} \right) + \sum_{h=1}^{q} (B^h - 1) + (q-1) \cdot P_{max}.$$

Evaluation of the maximum delay due to EF packets

We now consider the term $X_{EF} = a^1_{f(1)} - a^1_m + \sum_{h=1}^{q} (\sum_{g=f(h)}^{f(h+1)} C^h_{\tau(g)})$, that is the maximum delay due to EF packets and incurred by m. We denote *slow* the slowest node of line $\mathcal{L}$. That is for any flow τ_j visiting *slow*, for any node h visited by τ_j, we have $C^h_j \leq C^{slow}_j$.

We then distinguish the nodes visited by the EF flows before *slow* and those visited after. Thus, we have $\sum_{h=1}^{q}(\sum_{g=f(h)}^{f(h+1)} C_{\tau(g)}^{h})$ bounded by:

$$\underbrace{\sum_{h=1}^{slow-1}\left(\sum_{g=f(h)}^{f(h+1)-1} C_{\tau(g)}^{h} + C_{\tau(f(h+1))}^{h}\right)}_{\text{nodes visited before slow}} + \underbrace{\sum_{g=f(slow)}^{f(slow+1)} C_{\tau(g)}^{slow}}_{\text{node slow}} + \underbrace{\sum_{h=slow+1}^{q}\left(\sum_{g=f(h)+1}^{f(h+1)} C_{\tau(g)}^{h} + C_{\tau(f(h))}^{h}\right)}_{\text{nodes visited after slow}}.$$

For any node $h \in [1,q]$, for any EF packet m' visiting h, the processing time of m' on node h is less than $C_{\tau(m')}^{slow}$. Then, as packets are numbered consecutively from $f(1)$ to $f(q+1) = m$, we get inequation (1). By considering that on any node h, the processing time of an EF packet is less than or equal to: $C_{max}^{h} = \max_{j=1..n}(C_j^h)$, we get inequation (2).

$$\sum_{h=1}^{slow-1}\left(\sum_{g=f(h)}^{f(h+1)-1} C_{\tau(g)}^{h}\right) + \sum_{g=f(slow)}^{f(slow+1)} C_{\tau(g)}^{slow} + \sum_{h=slow+1}^{q}\left(\sum_{g=f(h)+1}^{f(h+1)} C_{\tau(g)}^{h}\right) \leq \sum_{g=f(1)}^{m} C_{\tau(g)}^{slow} \quad (1)$$

$$\sum_{h=1}^{slow-1} C_{\tau(f(h+1))}^{h} + \sum_{h=slow+1}^{q} C_{\tau(f(h))}^{h} \leq \sum_{\substack{h=1 \\ h\neq slow}}^{q} C_{max}^{h} \quad (2)$$

By inequations (1) and (2), we have: $X_{EF} \leq a_{f(1)}^{1} - a_m^1 + \sum_{g=f(1)}^{m} C_{\tau(g)}^{slow} + \sum_{\substack{h=1 \\ h\neq slow}}^{q} C_{max}^{h}$. The term $\sum_{g=f(1)}^{m} C_{\tau(g)}^{slow}$ is bounded by the maximum workload generated by the EF flows in the interval $[a_{f(1)}^{1}, a_m^1]$. By definition, this quantity is equal to:

$$\sum_{j=1}^{n}\left(1 + \left\lfloor \frac{a_m^1 - a_{f(1)}^1 + Jin_j^1}{T_j} \right\rfloor\right) \cdot C_j^{slow}.$$

As for any x, $\lfloor x \rfloor \leq x$, we can write: $\sum_{g=f(1)}^{m} C_{\tau(g)}^{slow} \leq \sum_{j=1}^{n}(1+(a_m^1 - a_{f(1)}^1 + Jin_j^1)/T_j) \cdot C_j^{slow}$. This bound is equal to: $\sum_{j=1}^{n}(1 + Jin_j^1/T_j) \cdot C_j^{slow} + (a_m^1 - a_{f(1)}^1) \cdot U_{EF}^{slow}$, where U_{EF}^{slow} denotes the processor utilization factor for the EF class on node *slow* and must be ≤ 1. Consequently, $X_{EF} \leq \sum_{j=1}^{n}(1 + Jin_j^1/T_j) \cdot C_j^{slow} + (a_m^1 - a_{f(1)}^1) \cdot (U_{EF}^{slow} - 1) + \sum_{\substack{h=1 \\ h\neq slow}}^{q} C_{max}^{h}$, that is bounded by: $\sum_{j=1}^{n}(1 + Jin_j^1/T_j) \cdot C_j^{slow} + \sum_{\substack{h=1 \\ h\neq slow}}^{q} C_{max}^{h}$.

Worst case end-to-end response time of an EF packet

We have shown in subsection 5.2 that $R_{max}{}_m^{\mathcal{L}}$, the worst case end-to-end response time of any packet m of flow τ_i, $i \in [1,n]$, is bounded by: $X_{EF} + \sum_{h=1}^{q}(B^h - 1) + (q-1) \cdot P_{max}$. Hence, considering the above defined bound for X_{EF}, we finally get:

Property 2 *When all the* EF *flows follow the same line $\mathcal{L}$ consisting of q nodes numbered from 1 to q, then the worst case end-to-end response time of any* EF *flow τ_i meets:*

$$R_{max}{}_i^{\mathcal{L}} \leq \sum_{j=1}^{n} \left(1 + \frac{Jin_j^1}{T_j}\right) \cdot C_j^{slow} + \sum_{\substack{h=1 \\ h \neq slow}}^{q} C_{max}^h + \sum_{h=1}^{q} (B^h - 1) + (q-1) \cdot P_{max}.$$

5.3.2 Extension

In the previous subsection, we assumed that all the EF flows followed the same line (i.e. the same sequence of nodes). In this subsection, we extend the obtained results to the general case, that is EF flows can follow different lines. Let us consider any EF flow $\tau_i \in \tau$ following line $\mathcal{L}$, that consists of q nodes numbered from 1 to q. To determine the worst case end-to-end response time of any packet m of τ_i, we adopt the same strategy as the one used in subsection 5.3.1, that consists in identifying the busy periods that affect the delay of m on the nodes visited by τ_i. For the sake of simplicity, we first consider that any EF flow τ_j following line $\mathcal{L}'$ with $\mathcal{L}' \neq \mathcal{L}$ and $\mathcal{L}' \bigcap \mathcal{L} \neq \emptyset$ never visits a node of line $\mathcal{L}$ after having left line $\mathcal{L}$ (cf. assumption 1). In subsection 5.3.3, we show how to remove this assumption.

Assumption 1 *For any* EF *flow τ_j following line $\mathcal{L}'$ with $\mathcal{L}' \neq \mathcal{L}$ and $\mathcal{L}' \bigcap \mathcal{L} \neq \emptyset$, if there exists a node $h \in \mathcal{L}' \bigcap \mathcal{L}$ such that τ_j visits $h' \neq h+1$ immediately after h, then τ_j never visits a node $h'' \in \mathcal{L}$ after.*

Decomposition of the end-to-end response time of m

Let us consider any packet m belonging to any EF flow $\tau_i \in \tau$ and following line $\mathcal{L}$, where $\mathcal{L}$ consists of q nodes numbered from 1 to q. Identifying the busy periods that affect the delay of m on the nodes visited by τ_i is somewhat more complex than in subsection 5.3.1, where all the EF flows followed the same line. Indeed, we still focus on the busy period bp^q in which m is processed on node q and we define $f(q)$ as the first EF packet, after the last of the packets not belonging to the EF class and processed in bp^q (if any). But contrary to the single line case, $f(q)$ does not necessarily come from node $q-1$ since the EF flow that $f(q)$ belongs to may follow a line different from $\mathcal{L}$.

Hence, to move backwards through the sequence of nodes m traverses, each time identifying preceding packets and busy periods that ultimately affect the delay of m, we have to consider an additional packet on node q, that is $p(q-1)$: the first EF packet processed between $f(q)$ and m on node q and coming from node $q-1$. We denote bp^{q-1} the busy period in which $p(q-1)$ has been processed on node $q-1$ and $f(q-1)$ the first EF packet processed in bp^{q-1} (after the last packet not belonging to the EF class). We then define $p(q-2)$ as the first EF packet processed between $f(q-1)$ and m on node $q-1$ and coming from node $q-2$. And so on until the busy period of node 1 in which the packet $f(1)$ is processed. We have thus determined the busy periods on nodes visited by τ_i that can be used to compute the end-to-end response time of packet m (cf. figure 4).

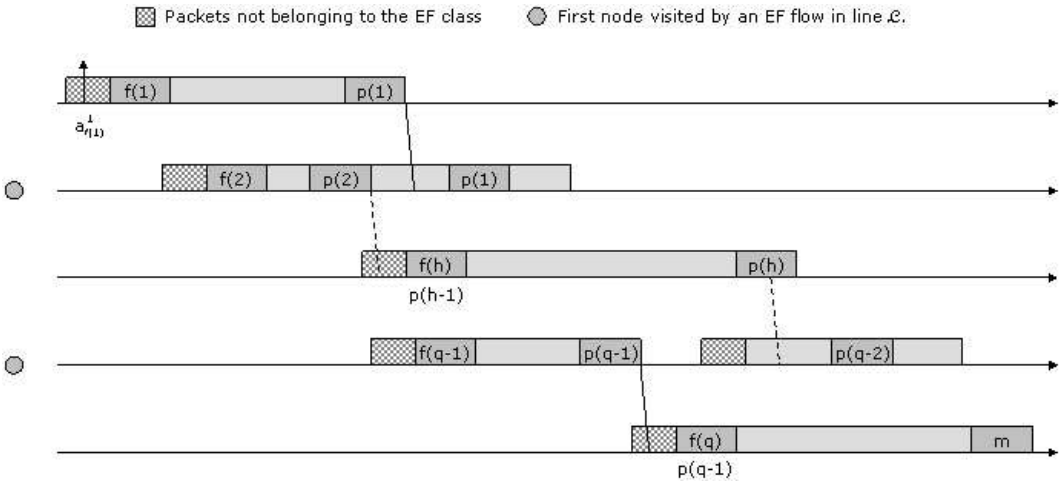

Figure 4: Response time of packet m

The latest departure time of packet m from node q is then equal to:

$a^1_{f(1)}$ + $(B^1 - 1)$ + the processing time on node 1 of packets $f(1)$ to $p(1)$ + $P^{1,2}_{p(1)}$
+ $(B^2 - 1)$ + the processing time on node 2 of packets $f(2)$ to $p(2)$ + $P^{2,3}_{p(2)}$ − $(a^2_{p(1)} - a^2_{f(2)})$
+ ...
+ $(B^q - 1)$ + the processing time on node q of packets $f(q)$ to m $-(a^q_{p(q-1)} - a^q_{f(q)})$.

Hence, the end-to-end response time of packet m is bounded by:

$$a^1_{f(1)} - a^1_m + \sum_{h=1}^{q} \left(\sum_{g=f(h)}^{p(h)} C^h_{\tau(g)} \right) + \sum_{h=1}^{q} (B^h - 1) + (q-1) \cdot P_{max} - \sum_{h=2}^{q} \left(a^h_{p(h-1)} - a^h_{f(h)} \right).$$

Let $first_j$ denote the first node visited by the EF flow τ_j in line $\mathcal{L}$. We can notice that on any node h of line $\mathcal{L}$, if there exists no EF flow τ_j such that $h = first_j$, then $p(h-1) = f(h)$ and so $a^h_{p(h-1)} - a^h_{f(h)} = 0$.

In other words, if $p(h-1) \neq f(h)$, then there exists an EF flow τ_j such that $h = first_j$. In such a case, by definition of $p(h)$, all the packets in $[f(h), p(h-1))$ cross line $\mathcal{L}$ for the first time at node h. We can then act on their arrival times. Postponing the arrivals of these messages in the busy period where $p(h-1)$ is processed, would increase the departure time of m from node q. Hence, in the worst case, $p(h-1) \in bp^h$ and $a^h_{p(h-1)} = a^h_{f(h)}$.

Thus, in the worst case, we obtain a decomposition similar to the one presented in subsection 5.3.1. By numbering consecutively on any node h the EF packets processed after $f(h)$ and before $p(h)$ (with $p(q) = m$), we get an upper bound on the end-to-end response time of packet m, that is:

$$a^1_{f(1)} - a^1_m + \sum_{h=1}^{q} \left(\sum_{g=f(h)}^{p(h)} C^h_{\tau(g)} \right) + \sum_{h=1}^{q} (B^h - 1) + (q-1) \cdot P_{max}.$$

Evaluation of the maximum delay due to EF packets

We now consider the term $X_{EF} = a^1_{f(1)} - a^1_m + \sum_{h=1}^{q} \left(\sum_{g=f(h)}^{p(h)} C^h_{\tau(g)} \right)$, that is the maximum delay due to EF packets and incurred by m. We denote *slow* the slowest node among the q nodes visited by m. That is for any flow τ_j visiting *slow*, for any node h visited by τ_j we have $C^h_j \leq C^{slow}_j$.

By definition, for any node $h \in [1, slow)$, $p(h)$ is the first packet belonging to the EF class, processed in bp^{h+1} and coming from node h. Moreover, $p(h)$ is the last packet considered in bp^h. Hence, if we count packets processed in bp^h and bp^{h+1}, only $p(h)$ is counted twice. In the same way, for any node $h \in (slow, q]$, $p(h-1)$ is the first packet belonging to the EF class, processed in bp^h and coming from node $h-1$. Moreover, $p(h-1)$ is the last EF packet considered in bp^{h-1}. Thus, $p(h-1)$ is the only packet counted twice when counting packets processed in bp^{h-1} and bp^h.

In addition, for any node $h \in [1, q]$, for any EF packet g visiting h, the processing time of g on node h is less than $C^{slow_{\tau(g)}}_{\tau(g)}$, where $slow_j$ is the slowest node visited by τ_j on line $\mathcal{L}$. Hence, if on any node $h \in [1, slow)$ (resp. $h \in (slow, q]$), we bound $p(h)$ (resp. $p(h-1)$) by $C^h_{max} = \max_{j \in \tau}(C^h_j)$, then we get:

$$X_{EF} \leq a^1_{f(1)} - a^1_m + \sum_{g=f(1)}^{m} C^{slow_{\tau(g)}}_{\tau(g)} + \sum_{\substack{h=1 \\ h \neq slow}}^{q} C^h_{max}.$$

We now evaluate the quantity $\sum_{g=f(1)}^{m} C_{\tau(g)}^{slow_{\tau(g)}}$. This quantity is bounded by the maximum workload generated by the EF flows visiting at least one node of line $\mathcal{L}$.

On node h, a packet of any EF flow τ_j visiting at least one node visited by τ_i can delay the execution of m if it arrives on node $first_j$, where $first_j$ denotes the first node common to τ_j and τ_i, at the earliest at time $a_{f(first_j)}^{first_j}$ and at the latest at time $a_m^{first_j}$. Indeed, if a packet of τ_j arrives on node $first_j$ after $a_m^{first_j}$, it will be processed on this node after m due to the FIFO-based scheduling of the EF class. So, if the next node visited by m is also the next node visited by the packet of τ_j, this packet will arrive after m on this node, and so on. Therefore, it will not delay packet m.

Thus, the packets of τ_j that can delay the execution of m are those arrived on node $first_j$ in the interval $[a_{f(first_j)}^{first_j} - Smax_m^{first_j} - Jin_j^1 \ , \ a_m^{first_j} - Smin_m^{first_j}]$, where $Smax_m^{first_j}$ (respectively $Smin_m^{first_j}$) denotes the maximum time (respectively the minimum time) taken by a packet of τ_j to arrive on node $first_j$. The maximum number of packets of τ_j that can delay m is then equal to: $1 + \lfloor (a_m^{first_j} - Smin_m^{first_j} - (a_{f(first_j)}^{first_j} - Smax_m^{first_j} - Jin_j^1)) \ / \ T_j \rfloor$.

By definition, $Smax_m^{first_j} - Smin_m^{first_j} + Jin_j^1 = Jin_j^{first_j}$, where $Jin_j^{first_j}$ denotes the delay jitter experienced by τ_j between its source node and node $first_j$. Applying this property to all the EF flows visiting at least one node visited by τ_i, we get:

$$\sum_{g=f(1)}^{m} C_{\tau(g)}^{slow_{\tau(g)}} \leq \sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \left(1 + \left\lfloor \frac{a_m^h - a_{f(h)}^h + Jin_j^h}{T_j} \right\rfloor \right) \cdot C_j^{slow_j} \right)$$

As $a_m^h - a_{f(h)}^h \leq a_m^h - a_m^1 + a_m^1 - a_{f(1)}^1$, we get $a_{f(1)}^1 - a_m^1 + \sum_{g=f(1)}^{m} C_{\tau(g)}^{slow_{\tau(g)}}$ bounded by:

$$a_{f(1)}^1 - a_m^1 + \sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \left(1 + \frac{a_m^h - a_m^1 + a_m^1 - a_{f(1)}^1 + Jin_j^h}{T_j} \right) \cdot C_j^{slow_j} \right), \text{ that is less than or equal to:}$$

$$\sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \left(1 + \frac{S_m^h + Jin_j^h}{T_j} \right) \cdot C_j^{slow_j} \right) + \left(a_m^1 - a_{f(1)}^1 \right) \cdot \left(\sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \frac{C_j^{slow_j}}{T_j} \right) - 1 \right).$$

Hence, if $\sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} (C_j^{slow_j} / T_j) \right) \leq 1$, the maximum delay due to EF packets and incurred by m is bounded by:

$$X_{EF} \leq \sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \left(1 + \frac{S_m^h + Jin_j^h}{T_j} \right) \cdot C_j^{slow_j} \right) + \sum_{\substack{h=1 \\ h \neq slow}}^{q} C_{max}^h.$$

Worst case end-to-end response time of an EF packet

In subsection 5.2, we have defined $Rmax_m^{\mathcal{L}}$ as the worst case end-to-end response time of any packet m of flow τ_i, $i \in [1,n]$, and shown that $Rmax_m^{\mathcal{L}} \leq X_{EF} + \sum_{h=1}^{q}(B^h - 1) + (q-1)\cdot Pmax$. Hence, considering the above defined bound for X_{EF}, we finally get:

Property 3 *If the condition* $\sum_{h=1}^{q}\left(\sum_{\substack{j=1\\first_j=h}}^{n}(C_j^{slow_j}/T_j)\right) \leq 1$ *is met, then the worst case end-to-end response time of any packet* m *of* EF *flow* τ_i *is bounded by:*

$$Rmax_i^{\mathcal{L}} \leq \sum_{h=1}^{q}\left(\sum_{\substack{j=1\\first_j=h}}^{n}\left(1+\frac{S_m^h + Jin_j^h}{T_j}\right)\cdot C_j^{slow_j}\right) + \sum_{\substack{h=1\\h\neq slow}}^{q} C_{max}^h + \sum_{h=1}^{q}(B^h-1) + (q-1)\cdot Pmax.$$

In section 6, we will see how to bound S_m^h, the time taken by packet m to arrive on node h, and Jin_j^h, the delay jitter experienced by τ_j between its source node and node h.

5.3.3 Generalization

Property 3 can be extended to the general case (i.e. by removing assumption 1). To achieve that, the idea is to consider any EF flow crossing line $\mathcal{L}$ after it left $\mathcal{L}$ as a new EF flow. We proceed by iteration until meeting assumption 1. We then apply Property 3 considering all these flows.

5.4 End-to-end jitter

We have assumed that any flow τ_i belonging to the EF class enters the DiffServ domain with a delay jitter bounded by Jin_i^1. But within the domain, the flow τ_i will experiment an additional jitter. To meet the constraint 2 given in Section 2, the last node visited by τ_i in the considered DiffServ domain has to hold every τ_i packet until its end-to-end response time is in the interval $[Rmax_i^{\mathcal{L}} - J\,, Rmax_i^{\mathcal{L}}]$, where $Rmax_i^{\mathcal{L}}$ is the worst case end-to-end response time of flow τ_i and J the worst case end-to-end delay jitter guarantee.

6 Admission control

We will now use the bound on the worst case response time for the EF flows to derive an admission control for the EF class. When a new flow belonging to the EF class arrives in the DiffServ domain, the admission control is in charge of deciding whether this flow can be accepted. The admission control has two major goals: (i) the new flow must not experience an end-to-end response time greater than the domain deadline, and (ii) the acceptance of the new flow must not compromise guarantees given to the already accepted EF flows.

Moreover, the design of an admission control for the EF class is a complex task in so far as the admission request of a new EF flow τ_i must not lead to recompute the worst case end-to-end response time of any EF flow already accepted. Indeed the acceptation of τ_i results in an increased sojourn time of τ_j, an already accepted EF flow crossing τ_i. This leads to an increased jitter of τ_j on the first node where τ_j crosses τ_k an already accepted EF flow not crossing τ_i. This in turn increases the sojourn time of τ_k, and so on.

To avoid this cascading effect, we define $d^h_{\mathcal{L}'}$ as the maximum sojourn time in node h guaranteed to any EF flow τ_j following $\mathcal{L}'$. The decomposition of the domain deadline D in intermediate deadlines is a general problem (see for instance [19]) and is outside the scope of this paper. The admission control uses this guaranteed sojourn time on node h to maximize the impact of any EF flow τ_j on τ_i. As long as the guaranteed sojourn time of τ_j is not violated, the guaranteed sojourn times as well as the end-to-end response time of any EF flow τ_k not crossing τ_i are not modified.

To reach its goals, the admission control has to verify the four following conditions when a new EF flow τ_i enters the DiffServ domain to follow a line $\mathcal{L}$. It is important to underline that only the worst case end-to-end response times experienced by EF flows visiting at least one node visited by τ_i have to be checked, and not those of all the EF flows in the DiffServ domain.

6.1 The local workload condition

When a new EF flow τ_i arrives in the DiffServ domain, the first condition the admission control must verify concerns the workload generated by all the EF flows on each node visited by τ_i. This condition is given in lemma 1.

Lemma 1 *A necessary condition for the feasibility of a set of EF flows on a node h, in the presence of flows of other classes, is $U^h_{EF} \leq 1$.*

6.2 The distributed workload condition

The admission control then checks that the condition required by property 3 holds, that is:

$$\sum_{h=1}^{q} \left(\sum_{\substack{j=1 \\ first_j = h}}^{n} \frac{C_j^{slow_j}}{T_j} \right) \leq 1.$$

6.3 The sojourn time condition

If the workload conditions are met, the admission control checks that on any node h visited by the new EF flow τ_i, this flow and each already accepted EF flow τ_j following $\mathcal{L}'$, with $h \in \mathcal{L}' \bigcap \mathcal{L}$, meet their maximum guaranteed sojourn time on this node, that is: $Rmax_i^h \leq d_{\mathcal{L}}^h$ and $Rmax_j^h \leq d_{\mathcal{L}'}^h$.

Lemma 2 *For any* EF *flow τ_i following line $\mathcal{L}$, for any node h visited by τ_i, the maximum sojourn time on node h of any packet of τ_i meets:*

$$Rmax_i^h \;\leq\; (B^h - 1) \;+ \sum_{\substack{\tau_j \text{ follows } \mathcal{L}' \\ h \in \mathcal{L}' \bigcap \mathcal{L}}} \left(1 + (Jin_j^h / T_j)\right) \cdot C_j^h .$$

Proof: The maximum sojourn time of m, a packet of the EF flow τ_i, consists of two parts. The first one is due to the non-preemption. As shown in subsection 5.2, this delay is less than or equal to $B^h - 1$. The second one is due to other EF packets arrived on node h before packet m. Then, we get:

$$\begin{aligned} Rmax_m^h &\leq a_{f(h)}^h - a_m^h + (B^h - 1) + \sum_{\substack{\tau_j \text{ follows } \mathcal{L}' \\ h \in \mathcal{L}' \bigcap \mathcal{L}}} \left(1 + \left\lfloor \frac{a_m^h - a_{f(h)}^h + Jin_j^h}{T_j} \right\rfloor \right) \cdot C_j^h \\ &\leq (B^h - 1) + \sum_{\substack{\tau_j \text{ follows } \mathcal{L}' \\ h \in \mathcal{L}' \bigcap \mathcal{L}}} \left(1 + \frac{Jin_j^h}{T_j}\right) \cdot C_j^h + \left(a_m^h - a_{f(h)}^h\right) \cdot \left(\sum_{j=1}^{n} \frac{C_j^h}{T_j} - 1\right). \end{aligned}$$

Moreover, $\sum_{j=1}^{n} (C_j^h / T_j) = U_{EF}^h \leq 1$ (see subsection 6.1 on the local workload condition). Hence the lemma. ■

6.4 The end-to-end response time condition

To meet the constraint 1 defined in subsection 2.4, the admission control finally checks that the end-to-end response times of τ_i and of each already accepted EF flow τ_j following line $\mathcal{L}'$ such that $\mathcal{L}' \bigcap \mathcal{L} \neq \emptyset$ meet the domain deadline D: $Rmax_i^{\mathcal{L}} \leq D$ and $Rmax_j^{\mathcal{L}'} \leq D$.

To compute the upper bound we have established in subsection 5.3.2 on the worst case end-to-end response time of any EF packet m, the admission control has to evaluate the terms Jin_j^h and S_m^h.

The former, Jin_j^h, that is the delay jitter experienced by flow τ_j between its source node and node h, is equal to: $Smax_j^h - Smin_j^h \leq \sum_{k=1}^{h-1}(d_{\mathcal{L}'}^k - C_j^k) + (h-1)\cdot(P_{max} - P_{min})$, where $\mathcal{L}'$ is the line followed by τ_j, composed of q' nodes numbered from 1 to q'.

The latter, S_m^h, that is the time taken by packet m to arrive on node h, is equal to: $a_m^h - a_m^1$. In the worst case, we have $a_m^h = d_m^{h-1} + P_{max}$. Then, we can write: $S_m^h \leq Rmax_m^{1,h-1} + P_{max}$, where $Rmax_m^{1,h-1}$ denotes the worst case response time of packet m when leaving node $h-1$. Hence, S_m^h is computed by iteration on the number of nodes in property 3.

7 Example

In this section, we give an example of computation of the bound on the end-to-end response time established in subsection 5.3.2. We assume that three EF flows τ_1, τ_2 and τ_3 have already been accepted. As shown in figure 5, τ_1 visits nodes 1, 2, 3 and 4, τ_2 visits nodes 5, 2, 3 and 6, and τ_3 visits nodes 7, 2 and 8.

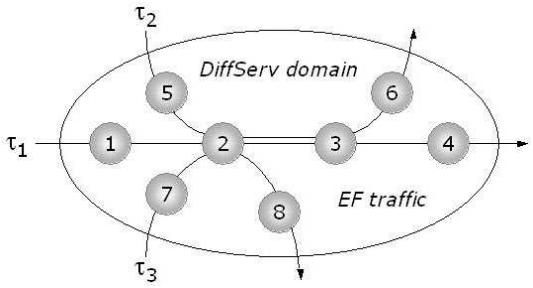

Figure 5: EF traffic in the DiffServ domain

We want to admit a new flow τ_4 visiting line $\mathcal{L}$ consisted of nodes 1 , 2, 3 and 4. We assume that the DiffServ domain deadline is $D = 60$. All the flows τ_i, $i = 1..4$, have a period $T_i = 10$ and a jitter $Jin_i^1 = 0$. For any packet of these EF flows, the processing time on a core router is equal to 2, the processing time on an ingress/egress router is equal to 3.

Moreover, we assume that flows not belonging to the EF class exist in the DiffServ domain. The maximum processing time of a packet of these flows are equal to 2 on a core router and 3 on an ingress/egress router. The maximum guaranteed sojourn time for any EF packet is equal to 9 on any node at the boundary of the network and 12 on any node in the core network. Finally, we have $P_{max} = P_{min} = 1$.

Assuming that the maximum sojourn time of any EF flow meets the local deadline, we can compute the end-to-end response time of the new EF flow τ_4. Indeed, as the distributed workload condition is met, that is $\sum_{h=1}^{4}(\sum_{\substack{j=1\\first_j=h}}^{4}(C_j^{slow_j}/T_j)) \leq 1$, we get $R_{max}{}_4^{\mathcal{L}}$ bounded by:

$$\underbrace{\sum_{h=1}^{4}\left(\sum_{\substack{j=1\\first_j=h}}^{4}\left(1+\frac{S_m^h+Jin_j^h}{T_j}\right)\cdot C_j^{slow_j}\right)}_{C_1^1+C_2^1+\left(1+\frac{15}{10}\right)\cdot C_3^2+\left(1+\frac{15}{10}\right)\cdot C_4^2} + \underbrace{\sum_{\substack{h=1\\h\neq slow}}^{q} C_{max}^h}_{7} + \underbrace{\sum_{h=1}^{q}(B^h-1)}_{6} + \underbrace{(q-1)\cdot P_{max}}_{3}.$$

The worst case end-to-end response time of the new EF flow is then bounded by 32.

As part of this article, a simulation tool has been developed for providing the exhaustive solution of a real-time scheduling problem in a network. Indeed, once the different parameters are specified, all the possible scenarios are generated and traffic feasibility is checked for each of them. The result of this simulation is a file containing the real worst case end-to-end response time of each flow and a scenario that leads to this response time. The real worst case end-to-end response time of the new EF flow, provided by the simulation tool, is equal to 26.

It is important to notice that if τ_2 or τ_3 are disturbed by a new EF flow not crossing τ_4, the real worst case end-to-end response time of τ_4 will possibly increase while the computed bound will remain the same. For example, if a new EF flow wants to go through the DiffServ domain by node 7, with a period equal to 20 and a processing time of any of its packets equal to 7, it will be accepted by the admission control. Then, the real worst case end-to-end response time of τ_4 will be equal to 28. If another EF flow having the same parameters wants to go through the DiffServ domain by node 5, it will also be accepted by the admission control. Then, the real worst case end-to-end response time of τ_4 will increase to 29, while the computed bound will remain equal to 32. Notice that the resources provisioned for the EF class that are not used are available for the other classes.

8 Conclusion

In this paper, we have proposed a solution to offer deterministic end-to-end real-time guarantees for flows in the EF class of the DiffServ model. The EF class is well adapted for flows with real-time constraints. We have considered sporadic flows that can model voice or video flows. We have determined a bound on the end-to-end response time of any sporadic flow with a trajectory analysis, less pessimistic than holistic analysis and we have compared this bound with the exact value provided by a simulation tool we have designed.

The MPLS technology reduces forwarding delays because of simpler processing and allows to indicate in a label the service class of the packet. We have shown how to implement our solution, based on the combination of DiffServ and MPLS. To accept a new EF flow in the DiffServ domain, we have proposed a simple admission control that involves only the EF flows crossing the new flow. Notice that even if the admission control uses the bounds we have established, the resources provisioned for the EF class that are not used are available for the other classes.

References

[1] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, T. Przygienda, *QoS routing mechanisms and OSPF extensions*, RFC 2676, August 1999.

[2] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, *RSVP-TE : Extensions to RSVP for LSP tunnels*, Internet draft, August 2001.

[3] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManusWang, *Requirements for traffic engineering over MPLS*, RFC 2702, September 1999.

[4] S. Baruah, R. Howell, L. Rosier, *Algorithms and complexity concerning the preemptive scheduling of periodic real-time tasks on one processor*, Real-Time Systems, 2, p 301-324, 1990.

[5] J. Bennett, K. Benson, A. Charny, W. Courtney, J. Le Boudec, *Delay jitter bounds and packet scale rate guarantee for Expedited Forwarding*, INFOCOM'2001, Anchorage, USA, April 2001.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, *An architecture for Differentiated Services*, RFC 2475, December 1998.

[7] R. Braden, D. Clark, S. Shenker, *Integrated services in the Internet architecture: an overview*, RFC 1633, June 1994.

[8] A. Charny, J. Le Boudec, *Delay bounds in a network with aggregate scheduling*, QoFIS, Berlin, Germany, October 2000.

[9] F. Chiussi, V. Sivaraman, *Achieving high utilization in guaranteed services networks using early-deadline-first scheduling*, IWQoS'98, Napo, California, May 1998.

[10] L. Georgiadis, R. Guérin, V. Peris, K. Sivarajan, *Efficient network QoS provisioning based on per node traffic shaping*, IEEE/ACM Transactions on Networking, Vol. 4, No. 4, August 1996.

[11] L. George, S. Kamoun, P. Minet, *First come first served: some results for real-time scheduling*, PDCS'01, Dallas, Texas, August 2001.

[12] L. George, D. Marinca, P. Minet, *A solution for a deterministic QoS in multimedia systems*, International Journal on Computer and Information Science, Vol.1, N3, September 2001.

[13] M. Gerla, C. Casetti, S. Lee, G. Reali, *Resource allocation and admission control styles on QoS DiffServ networks*, QoS-IP 2001, Rome, Italy, 2001.

[14] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, *Assured Forwarding PHB group*, RFC 2597, June 1999.

[15] V. Jacobson, K. Nichols, K. Poduri, *An Expedited Forwarding PHB*, RFC 2598, June 1999.

[16] J. Le Boudec, P. Thiran, *A note on time and space methods in network calculus*, Technical Report, No. 97/224, Ecole Polytechnique Fédérale de Lausanne, Swiss, April 11, 1997.

[17] F. Le Faucheur, L. Wu, S. Davari, et al. *MPLS support of Differentiated Services*, Internet draft, August 2000.

[18] A. Parekh, R. Gallager, *A generalized processor sharing approach to flow control in integrated services networks: the multiple node case*, IEEE ACM Transactions on Networking, Vol.2, N2, April 1994.

[19] D. Raz, Y. Shavitt, *Optimal partition of QoS requirements with discrete cost functions*, INFOCOM'2000, Tel-Aviv, Israel, March 2000.

[20] V. Sivaraman, F. Chiussi, M. Gerla, *Traffic shaping for end-to-end delay guarantees with EDF scheduling*, IWQoS'2000, Pittsburgh, June 2000.

[21] V. Sivaraman, F. Chiussi, M. Gerla, *End-to-end statistical delay service under GPS and EDF scheduling: a comparaison study*, INFOCOM'2001, Anchorage, Alaska, April 2001.

[22] K. Tindell, J. Clark, *Holistic schedulability analysis for distributed hard real-time systems*, Microprocessors and Microprogramming, Euromicro Journal, Vol. 40, pp. 117-134, 1994.

[23] M. Vojnovic, J. Le Boudec, *Stochastic analysis of some expedited forwarding networks*, INFOCOM'2002, New York, June 2002.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399